**### YOLO\_V3 실습 (최소메모리요구사항: 2,048MB) ###**

>>> **VirtualBox** >>> **USB** >>> **Webcam (설정)**

**01. Anaconda 가상환경 생성**

# conda create -y --name **yolo** python=3.9

# conda env list

**02. Anaconda 가상환경 실행**

# conda activate **yolo**

**03. YOLOv3.zip 다운로드 및 Unzip**

>>> **YOLOv3.zip** 다운로드 >>> **/root** 디렉토리로 이동 (마우스 오른쪽버튼 클릭 >>> 다른 위치로 이동…)

>>> **Unzip YOLOv3.zip** (마우스 오른쪽버튼 클릭 >>> 여기에 풀기)

**04. YOLOv3 디렉토리로 이동**

# pwd

# cd YOLOv3.zip

**05. Python Package 설치**

# cat requirements.txt

=========================================================

flask==2.2.2

numpy>=1.18.2

scipy>=1.4.1

wget>=3.2

seaborn>=0.10.0

tensorflow==2.9.1

opencv-python==4.4.0.46

tqdm==4.43.0

pandas

awscli

urllib3

mss

=========================================================

>>> requirements.txt 파일 확인

# pip install -r requirements.txt

**06. app.py 실행**

# python app.py

**07. Chrome 브라우저 실행**

>>> **http://127.0.0.1:5000**

**### YOLO\_V3 Configuration ###**

**01. Model Data**

>>> model\_data/**yolov3-tiny-custom\_final.weights**

>>> model\_data/**mask/mask.names**

**02. yolov3/configs.py 수정**

===================================================================================

YOLO\_V3\_TINY\_WEIGHTS = "model\_data/yolov3-tiny-custom\_final.weights"

YOLO\_COCO\_CLASSES = "model\_data/mask/mask.names"

YOLO\_INPUT\_SIZE = 416

TRAIN\_YOLO\_TINY = True

===================================================================================

**03. yolov3/utils.py 스트리밍 함수 추가 (웹 스트리밍시)**

def detect\_realtime\_image(Yolo, image, input\_size=416, CLASSES=YOLO\_COCO\_CLASSES,

score\_threshold=0.3, iou\_threshold=0.45, rectangle\_colors=''):

original\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

image\_data = image\_preprocess(np.copy(original\_image), [input\_size, input\_size])

image\_data = image\_data[np.newaxis, ...].astype(np.float32)

print(image\_data.shape)

if YOLO\_FRAMEWORK == "tf":

pred\_bbox = Yolo.predict(image\_data)

print(pred\_bbox)

elif YOLO\_FRAMEWORK == "trt":

batched\_input = tf.constant(image\_data)

result = Yolo(batched\_input)

pred\_bbox = []

for key, value in result.items():

value = value.numpy()

pred\_bbox.append(value)

pred\_bbox = [tf.reshape(x, (-1, tf.shape(x)[-1])) for x in pred\_bbox]

pred\_bbox = tf.concat(pred\_bbox, axis=0)

bboxes = postprocess\_boxes(pred\_bbox, original\_image, input\_size, score\_threshold)

bboxes = nms(bboxes, iou\_threshold, method='nms')

image = draw\_bbox(original\_image, bboxes, CLASSES=CLASSES, rectangle\_colors=rectangle\_colors)

# CreateXMLfile("XML\_Detections", str(int(time.time())), original\_image, bboxes, read\_class\_names(CLASSES))

return image

**04. detection\_custom.py 수정 및 파일 이동**

from yolov3.utils import Load\_Yolo\_model, detect\_realtime\_image

image\_path = "./IMAGES/images19.jpg"

yolo = Load\_Yolo\_model()

img = cv2.imread(image\_path)

predimg = detect\_realtime\_image(yolo, img)

predimg = cv2.cvtColor(predimg, cv2.COLOR\_BGR2RGB)

cv2.imshow('pred', predimg )

cv2.waitKey(0)

cv2.destroyAllWindows()

**05. templates 폴더에 base.html 및 index.html작성**

>>> **base.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Mask Detect</title>

<link rel="preconnect" href="https://fonts.googleapis.com">

<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<link href="https://fonts.googleapis.com/css2?family=Nanum+Gothic&family=Noto+Sans+KR:wght@300&display=swap" rel="stylesheet">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3" crossorigin="anonymous">

</head>

<body>

<nav class="navbar bg-light border-bottom">

<div class="container-fluid">

<a class="navbar-brand bg-" href="/">Mask Detect</a>

</div>

</nav>

<div class="container">

{% block content %}{% endblock %}

</div>

<footer class="text-center">

{% block footer %}

<small> YOLO\_V3 Test </small>

{% endblock %}

</footer>

</body>

</html>

>>> **index.html**

{% extends 'base.html' %}

{% block content %}

<video id="video" width="0" height="0" autoplay style="background-color: grey"></video>

<table>

<tr>

<td>WebCam<br>before detect mask</td>

<td>Detect<br>after detect mask</td>

</tr>

<tr>

<td><canvas id="canvas" width="416" height="416" style="background-color: grey"></canvas></td>

<td><img id="image" src="" width="416" height="416" style="background-color: grey"></img></td>

</tr>

</table>

<script>

var video = document.getElementById('video');

var canvas = document.getElementById('canvas');

var context = canvas.getContext('2d');

var image = document.getElementById('image');

// Get access to the camera!

if(navigator.mediaDevices && navigator.mediaDevices.getUserMedia) {

navigator.mediaDevices.getUserMedia({ video: true }).then(function(stream) {

video.srcObject = stream;

video.play();

window.setInterval(function() {

context.drawImage(video, 0, 0, 416, 416); // better use size because camera may gives data in different size then <video> is displaying

canvas.toBlob(upload, "image/jpeg");

}, 100);

});

}

function upload(file) {

var formdata = new FormData();

formdata.append("snap", file);

var xhr = new XMLHttpRequest();

xhr.open("POST", "{{ url\_for('upload') }}", true);

xhr.responseType = 'blob';

xhr.onload = function() {

if(this.status = 200) {

//console.log(this.response);

} else {

console.error(xhr);

}

image.src = URL.createObjectURL(this.response); // blob

};

// send form in AJAX

xhr.send(formdata);

//image.src = URL.createObjectURL(file);

}

</script>

{% endblock %}

**06. app.py**

from flask import Flask, request, make\_response, render\_template, redirect, url\_for, session

import cv2

import numpy as np

import datetime

import os

os.environ['CUDA\_VISIBLE\_DEVICES'] = '0'

from yolov3.utils import Load\_Yolo\_model, detect\_realtime\_image

from yolov3.configs import \*

yolo = Load\_Yolo\_model()

app = Flask(\_\_name\_\_)

app.config['SECRET\_KEY'] = 'aiot'

@app.before\_request

def before\_request():

session.permanent = True

app.permanent\_session\_lifetime = datetime.timedelta(minutes=5)

session.modified = True

@app.route("/", methods=['POST', 'GET'])

def index():

return render\_template('index.html')

def send\_file\_data(data, mimetype='image/jpeg', filename='output.jpg'):

response = make\_response(data)

response.headers.set('Content-Type', mimetype)

response.headers.set('Content-Disposition', 'attachment', filename=filename)

return response

@app.route('/upload', methods=['GET', 'POST'])

def upload():

if request.method == 'POST':

fs = request.files.get('snap')

if fs:

img = cv2.imdecode(np.frombuffer(fs.read(), np.uint8), cv2.IMREAD\_UNCHANGED)

img = detect\_realtime\_image(yolo, img)

img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

ret, buf = cv2.imencode('.jpg', img)

return send\_file\_data(buf.tobytes())

else:

return 'You forgot Snap!'

return redirect(url\_for('index'))

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True, host='0.0.0.0', port=5000)

* **수고하셨습니다 -**